جلسه پنجم: پیاده‌سازی اولیه و توسعه سیستم UTAXI

جلسه پنجم پروژه **UTAXI** یکی از مهم‌ترین نقاط عطف این پروژه محسوب می‌شود، زیرا در این مرحله تیم وارد **فاز پیاده‌سازی کدها** شد. پس از هفته‌ها تحلیل نیازمندی‌ها، طراحی اولیه سیستم، بررسی چالش‌ها و برنامه‌ریزی برای اجرای دقیق فازهای پروژه، اکنون زمان آن رسیده بود که تیم اولین نسخه از **کدهای اصلی سیستم** را پیاده‌سازی کند. انتخاب زبان برنامه‌نویسی **C++** برای این مرحله از پروژه، باعث شد که تیم نیاز به هماهنگی بالایی در تعریف ساختارهای اصلی سیستم داشته باشد و به **استانداردسازی نحوه کدنویسی** نیز توجه ویژه‌ای کند.

جلسه با مدیریت **امین یوسفی** آغاز شد و او در ابتدا **مروری کلی بر جلسات قبلی** انجام داد تا اعضای تیم درک بهتری از مسیر طی‌شده و گام‌های پیش رو داشته باشند. او توضیح داد که تیم تاکنون روی **تحلیل نیازها، طراحی معماری سیستم، برنامه‌ریزی پروژه با Microsoft Project و تعریف فرایندهای اصلی** تمرکز داشته و اکنون **باید تمرکز را بر روی پیاده‌سازی واقعی سیستم بگذاریم**.

یوسفی تأکید کرد که یکی از **مهم‌ترین چالش‌های پیش‌روی تیم**، حفظ هماهنگی بین اعضا در زمان نوشتن کدها است. هر عضو تیم مسئول توسعه بخشی از سیستم خواهد بود و اگر استاندارد مشخصی برای نحوه کدنویسی، نام‌گذاری متغیرها و نحوه ذخیره‌سازی داده‌ها تعیین نشود، در ادامه پروژه دچار مشکلات زیادی خواهند شد. از این رو، قبل از شروع پیاده‌سازی، تصمیم گرفته شد که تیم **استانداردهای مشخصی برای توسعه کدها تعیین کند** و پس از آن، اعضا شروع به نوشتن کلاس‌های اصلی کنند.

بخش اول: تعیین استانداردهای کدنویسی و روش‌های همکاری در تیم

برای جلوگیری از بروز مشکلات ناشی از ناسازگاری در شیوه کدنویسی، تیم تصمیم گرفت که **یک سری استانداردهای مشخص و یکسان** برای تمامی اعضا تعیین کند. این استانداردها شامل موارد زیر بود:

✅ **استفاده از CamelCase** برای نام‌گذاری متغیرها و توابع، تا کد خواناتر و منسجم‌تر باشد.
✅ **تعریف هر کلاس در یک فایل .h) Header) و پیاده‌سازی آن در یک فایل .cpp) Source)**، تا ساختار پروژه مرتب و خوانا باقی بماند.
✅ **استفاده از الگوهای طراحی شی‌گرا (Object-Oriented Design)** برای جلوگیری از تکرار کدها و افزایش انسجام سیستم.
✅ **افزودن کامنت‌های واضح در تمامی کلاس‌ها و توابع**، تا در آینده، توسعه‌دهندگان دیگر نیز بتوانند به‌راحتی کدها را درک کنند.
✅ **اجرای تست‌های محلی قبل از ارسال تغییرات به GitHub**، تا از بروز خطاهای کامپایل و مشکلات احتمالی جلوگیری شود.
✅ **مدیریت نسخه‌ها و هماهنگی در GitHub**، تا تیم بتواند تغییرات را به‌درستی کنترل کند و از بروز مشکلات مرتبط با **تعارض در نسخه‌ها (Merge Conflicts)** جلوگیری شود.

این استانداردها در طول جلسه توسط تمامی اعضای تیم تأیید و مورد بحث قرار گرفت و به عنوان **چارچوب اصلی کدنویسی پروژه** به کار گرفته شد.

بخش دوم: تقسیم وظایف بین اعضای تیم

پس از مشخص شدن استانداردها، تیم وارد **مرحله تقسیم وظایف** شد. از آنجایی که هر عضو تیم تخصص و مهارت خاصی در حوزه‌ای مشخص داشت، وظایف به‌گونه‌ای تخصیص داده شد که بیشترین بهره‌وری حاصل شود. تقسیم‌بندی وظایف به شرح زیر انجام شد:

- **مبینا مهرآذر**: مسئول طراحی و پیاده‌سازی **کلاس DatabaseManager** که وظیفه **ذخیره اطلاعات کاربران و سفرها** را بر عهده دارد.
- **محمدرضا نعمتی**: مسئول توسعه **کلاس BackendController** که ارتباطات سرور و APIها را مدیریت می‌کند.
- **آرین باستانی**: مسئول طراحی و پیاده‌سازی **رابط کاربری کنسول (CLI)** و ایجاد تعاملات اولیه بین کاربران و سیستم.
- **امین یوسفی**: مدیر پروژه، مسئول **نظارت بر روند پیاده‌سازی** و **هماهنگی بین اعضای تیم**.
- **محمد امانلو**: مسئول تست و تضمین کیفیت کدها، اجرای تست‌های اولیه و **مدیریت خطاها**.

این تقسیم وظایف باعث شد که هر عضو تیم وظایف مشخصی داشته باشد و بتواند به‌طور دقیق روی بخش خود کار کند.

بخش سوم: پیاده‌سازی کلاس‌های اصلی سیستم

با تعیین استانداردها و مشخص شدن وظایف، تیم وارد فاز **کدنویسی واقعی** شد. در این مرحله، کلاس‌های اصلی سیستم پیاده‌سازی شدند.

۱. کلاس User (کلاس پایه برای کاربران)

این کلاس ویژگی‌های مشترک کاربران (مسافران و رانندگان) را شامل می‌شود:

```
class User {

public:

    string name;

    string email;

    string phoneNumber;

    int userID;



    void registerUser();

    void updateUserDetails();

};
```

---

۲. کلاس Driver (مخصوص رانندگان)

کلاس Driver از User ارث‌بری کرده و ویژگی‌های خاص رانندگان را شامل می‌شود:

```
class Driver : public User {
public:
    string vehicleDetails;
    int driverRating;

    void acceptRide();
    void updateDriverAvailability();
};
```

---

۳. کلاس Passenger (مخصوص مسافران)

کلاس Passenger نیز از User ارث‌بری کرده و متدهایی برای درخواست سفر دارد:

```
class Passenger : public User {
public:
    void requestRide();
    void rateDriver();
};
```

---

۴. کلاس RideRequest (مدیریت درخواست‌های سفر)

این کلاس مسئول **مدیریت درخواست‌های سفر** است:

```
class RideRequest {

public:

    int rideID;

    string pickupLocation;

    string dropoffLocation;

    int passengerID;

    int driverID;



    void createRide();

    void cancelRide();

};
```

---

۵. کلاس SystemManager (مدیر اصلی سیستم)

این کلاس مدیریت کلی سیستم را انجام می‌دهد:

```
class SystemManager {

public:

    void initializeSystem();

    void handleRideRequests();

};
```

بخش چهارم: بررسی چالش‌های پیش رو و راه‌حل‌های ارائه‌شده

در فرآیند توسعه اولیه، چالش‌های متعددی مطرح شد که برخی از آن‌ها عبارت بودند از:

1️⃣**مشکلات هماهنگی در GitHub**: برخی اعضا در ادغام تغییرات دچار **مشکلات نسخه‌ای و Merge Conflict** شدند.
✅ **راه‌حل**: آموزش کوتاهی در مورد نحوه مدیریت **Branch ها** و روش صحیح Merge تغییرات ارائه شد.

2️⃣**وابستگی زیاد کلاس‌ها به یکدیگر**: برخی کلاس‌ها به‌شدت به یکدیگر وابسته بودند که باعث می‌شد پیاده‌سازی مستقل آن‌ها دشوار شود.
✅ **راه‌حل**: بازنگری در معماری کد و استفاده از **Dependency Injection** برای کاهش وابستگی‌های مستقیم.

3️⃣**بروز خطاهای کامپایل**: برخی اعضا هنگام اجرای کدها دچار مشکلات کامپایل شدند.
✅ **راه‌حل**: تعریف مرحله **Code Review** قبل از Commit نهایی در GitHub.

بخش پنجم: برنامه‌ریزی برای جلسات آینده

پس از پایان این جلسه، تیم تصمیم گرفت که در جلسات بعدی روی موارد زیر تمرکز کند:

- **تکمیل پیاده‌سازی کلاس‌ها و افزودن ویژگی‌های پیشرفته**
- **نوشتن تست‌های جامع برای بررسی عملکرد صحیح سیستم**
- **اجرای تست‌های کامل روی کدهای نهایی و ادغام آن‌ها**
- **برنامه‌ریزی برای پیاده‌سازی قابلیت‌های پیشرفته شامل سیستم امتیازدهی و محاسبه هزینه سفر**

نتیجه‌گیری

این جلسه یکی از **مهم‌ترین جلسات پروژه** محسوب شد و تیم موفق شد **اولین نسخه از کدهای سیستم را پیاده‌سازی کند**. با این پیشرفت، تیم برای توسعه ویژگی‌های جدید آماده شد و هماهنگی بیشتری بین اعضا ایجاد شد. در جلسات آینده، تمرکز اصلی بر روی **تست و بهینه‌سازی سیستم خواهد بود**.